開放型仮想ショップフロアの開発（第2報）*

—— 並列分散シミュレーションシステムの実装 ——

尾 崎 優** 小野里 雅彦*** 岩 田 一 明†

Development of Virtual Shop Floor Based on Open System Architecture (2nd Report)

— Implementation of Distributed and Parallel Simulation System —

Suguru OSAKI, Masahiko ONOSATO and Kazuaki IWATA

“Virtual Manufacturing” is one of the most attractive techniques in the field of manufacturing. In order to make a Virtual Shop Floor (VSF) as a first step to build Virtual Manufacturing Systems, Open System Architecture for Virtual Shop floor (OSAViS) is presented. OSAViS is an architecture to make the VSF as a set of various modules and execute a distributed and parallel simulation of shop floors. In this paper, a data management method, a synchronization process, and a communication mechanism in OSAViS are described. These subjects are important to simulate shop floors on a distributed and parallel environment. Based on OSAViS, a pilot system named Open-VirtualWorks is being developed so as to evaluate the validity of OSAViS. The current system can simulate behaviors of automated guided vehicles on a shop floor. By modeling and simulating with this system, a validation of the data management method, synchronization, and the communication mechanism is done, and the effectiveness of OSAViS is shown.

Key words: virtual manufacturing, open system, distributed parallel simulation, synchronization, data management

1. はじめに

近年、生産システムの計算機による事前評価の手段として、仮想生産 (Virtual Manufacturing) の概念に関心が集まっている¹⁾。そこで著者らは、ショップフロアレベルでの仮想生産を実現するための仕組みである仮想ショップフロア (Virtual Shop Floor, VSF) に関して研究、開発を行ってきた²⁾。

仮想ショップフロアの実用化には、ショップフロア全体までを対象にした大規模なシステムを効率的に開発する手法と現実的な時間応答性を有する処理手法が求められる。著者らはこうした課題に対して、仮想ショップフロアをモジュール化システムとして構築、運用するためのシステムアーキテクチャ、OSAViS (Open System Architecture for Virtual Shop floor) の概念を示している³⁾。

本論文では、まず、OSAViS の基本概念について説明し、その概念に基づき並列分散シミュレーションを実現する上で重要となるデータ管理、同期管理およびモジュール間コミュニケーションの手法について説明する。さらに、OSAViS の有効性を検証するために実装されたシステムである Open-VirtualWorks について説明し、そこで実行される並列分散シミュレーション結果を示す。

2. 開放型仮想ショップフロアの基本概念

本節では、前報⁴⁾で述べた開放型仮想ショップフロアの基本概念と OSAViS の枠組みについて簡単にまとめる。

仮想ショップフロアの実現のためには、ショップフロアの状態を表現し、かつ、シミュレーションによりその状態を更新するショップフロアモデル機能と、シミュレーションを整合的、効率的に実現するためのシミュレーション管理機能を実装する必要がある。OSAViS では、VSF モジュールとよばれる要素を組み合わせるこ

とで仮想ショップフロアを構築する。また、以上の機能は VSF モジュール間でのメッセージ交換を介して提供され、その内容を VSF サービスとよぶ。OSAViS では、複数の VSF モジュールが互いに VSF サービスの授受を行うことで、全体としてショップフロアの並列分散シミュレーションを実現する。

そこで本報では、前報で述べた OSAViS に基づく仮想ショップフロアを具体的に実装する上で重要な、公開データの管理手法、シミュレーションの同期管理手法、および、VSF モジュール間でのコミュニケーションメカニズムについて説明する。

3. OSAViS における公開データ管理

OSAViS では、ショップフロアモデル機能を提供するために利用されるデータをシミュレーションデータとよぶ。シミュレーションデータはその管理方法の違いにより「内部データ」と「公開データ」に分類される。内部データとは、それぞれの VSF モジュールが個別にその内部で利用するデータであり、参照や変更などの処理はデータを保有する VSF モジュールが自らの責任において実行する。また、公開データとは、データを所有する VSF モジュールの外部に存在し、他の VSF モジュールによる自由な参照が認められたデータである。本節では、OSAViS における公開データの管理手法について説明する。

3.1 公開データとデータオブジェクト

OSAViS において、ショップフロアの状態は複数のエンティティモデルを組み合わせることで表現される。エンティティモデルは、ショップフロアを構成する「もの (entity)」に着目しモデル化したものであり、「もの」の状態を表す属性を持ち、能動的に属性値の更新を行う状態更新プロセスを有する。

OSAViS におけるシミュレーションデータとはエンティティモデルの属性値を意味する。また、シミュレーションデータにおける公開データとは、エンティティモデルが実装されている VSF モ

* 原稿受付 平成 10 年 6 月 5 日

** 学生会員 大阪大学大学院工学研究科 (吹田市山田丘 2-1)

*** 正会員 大阪大学大学院工学研究科

† 正会員 高知工業高等専門学校 (南国市物部乙 200-1)

ジュールの外部にその属性値の一部を置くことで、他のモジュールによる自由な参照を認めたものである。したがって、公開データの変更はそのデータに関するエンティティモデルが実装されたVSFモジュールが所有者モジュールとなっていくが、参照に関する処理は、所有者モジュールは関与しない。こうした状況で公開データを利用する場合、以下に示す管理の必要性が生じる。

- (1) データ変更におけるデータごとの所有者の確認
- (2) 変更前のデータを参照していたVSFモジュールに対する変更の通知

本研究では、データオブジェクトを導入することで、個々のデータについて個別にその状態を把握し、それぞれのデータに応じて細やかな管理を行う。データオブジェクトとは、公開データである「データコンテンツ」に、データオブジェクトを管理するために必要な属性である「データオブジェクト属性」およびデータコンテンツに様々な処理を施すための手続きである「処理メソッド」を付加し、一つのパッケージとしたものである。また、データオブジェクトはタイプを持ち、同一タイプのデータオブジェクトは、同種のデータコンテンツと同じ処理メソッドを持つ。

また、各々のデータオブジェクトはその属性の一つとして唯一の名前を持っており、データオブジェクトマネージャが名前によりそれらを区別、管理する。データオブジェクトマネージャとは、様々なVSFモジュールからのメッセージを適切なデータオブジェクトに対して転送し、必要に応じてその結果を返送するというように、VSFモジュールとデータオブジェクトの間でのメッセージの配送を行うものである。

3.2 データコンテンツの参照と変更

データコンテンツの参照や変更に関する処理は、データオブジェクトの処理メソッドを実行することで行われる。データを利用したいVSFモジュールは直接データにアクセスするのではなく、データオブジェクトに対して処理メソッドの実行を要求するメッセージを送信することで参照や変更を行う。

データを参照したVSFモジュールはデータオブジェクト属性の一つであるユーザ属性に登録される。参照されたデータに変更が生じた場合には、登録されているVSFモジュールに対して、データの変更が通知される。また、データの変更に関しては、データオブジェクト属性に所有者属性があり、そこに登録されているVSFモジュールにのみ変更を許す。データの変更と参照に関する一連の処理手続きを図1に示す。

このように、個々のデータに属性を付加することでデータごとの細かな管理が可能となる。また、データに対する処理を処理メソッドという形でデータに付加することで、データ変更の通知など、参照や変更にもともなう生じる様々な付加的処理をデータごとに行うことができる。データオブジェクトを利用することで、多数のVSFモジュールによるデータの利用を統合的に管理することが可能となる。

4. OSAViSにおける同期管理

一般に、並列分散シミュレーション環境においては、ハードウェア性能や処理内容などの条件により、並列プロセス間での処理時間にばらつきが生じる。したがって、シミュレーション全体としての時間的な整合性を保つため、並列プロセス間での同期管理を行う必要がある⁹⁾。しかしながら、厳密な同期管理を行うとVSFモジュールに対する制約が強くなり、VSFモジュールの処

Fig.1 Procedure of change and reference of data

理の進行における自由度を減らすこととなる。そこで本節では、個々のVSFモジュールの処理の自由度を考慮したOSAViSにおける同期管理の仕組みについて述べる。

4.1 同期管理のための時間軸

OSAViSでは、シミュレーションの進行を管理するため、基準時間軸(Standard Time Axis)と個別時間軸(Individual Time Axis)の2種類の時間軸が定義されている。

基準時間軸は、構築した仮想ショップフロアに唯一に存在し、シミュレーション全体の基準となる。基準時間軸は実世界における連続した時間軸を複数の区間に分割することで離散化を行った時間軸である。OSAViSでは、この分割された時間を基準時間ユニット(Standard Time Unit)とよび、 $U_n(t_n, \alpha_n)$ で表す。ここで、 n はユニット番号、 t_n はユニットの開始時刻、 α_n はユニットの時間幅である。OSAViSにおいては、ユニット番号の更新によりシミュレーションが進行する。

また、個別時間軸とは各々のVSFモジュールが内部で個別に持っている時間軸である。VSFモジュールは、離散連続といった時間軸の種類や、離散時間軸におけるステップ幅などの時間軸の詳細を自由に個別時間軸として定義し、その時間軸に基づき内部での処理を実行できる。ただし、VSFモジュールが利用する個別時間軸における時刻は、基準時間軸における基準時間ユニットと対応づけられなければならない。次項では、以上の2種類の時間軸を利用した同期管理のプロセスについて説明する。

4.2 OSAViSにおける同期管理プロセス

基準時間ユニットには、シミュレーションの時系列にそって、シミュレーション結果の確定した過去基準時間ユニット(Past Standard Time Unit)、結果の確定のためにVSFモジュール間でVSFサービスの授受を行っている現在基準時間ユニット(Current Standard Time Unit)、結果の確定していない未来基準時間ユニット(Future Standard Time Unit)がある(図2参照)。OSAViSでは、シミュレーションの同期管理のための機能と基準時間軸を保有するVSFモジュールが存在し、これを同期管理モジュールとよぶ。同期管理モジュールは他のVSFモジュールの処理の進行状況を

Fig.2 Relation of time axes

把握し、現在基準時間ユニットにおける処理が終了しシミュレーション結果が確定すると、現在基準時間ユニットを更新する。以下に、同期管理モジュールによる処理の流れを示す。

- (1) 同期管理モジュールが他の VSF モジュールに対して、現在基準時間ユニット $U_n(t_n, \alpha_n)$ を通知する。
- (2) 各々の VSF モジュールは、 U_n におけるシミュレーション結果を確定させるために、相互に VSF サービスの授受を行う。
- (3) 結果の確定した VSF モジュールは、その処理の完了を同期管理モジュールに対して知らせる。
- (4) 同期管理モジュールは、VSF モジュールによる処理の完了を確認し、現在基準時間ユニットを U_{n+1} に更新し、それを VSF モジュールに対して通知する。

個々の VSF モジュールは、内部的に未来基準時間ユニットにおけるシミュレーションを先行して実行することで、あらかじめ暫定的な結果を計算しておくことができる。ただし、VSF モジュール間でやり取りされる情報は、現在基準時間ユニットにおける情報に限定される。

また、OSAViS では、同一の基準時間ユニット内で生じた事象は、その前後関係を考慮しない。これは、個々の VSF モジュールによる自由な個別時間軸の設定と利用を認めているため、厳密に事象の時間的前後関係を考慮する場合、それぞれの VSF モジュールのシミュレーションの進行に対する管理が厳しくなり、分散処理による利点が発揮できなくなるためである。そこで本手法では、シミュレーションの応答性と精度に関する要求に対しては、ユニット幅 α_n を変えることで対応する。

以上の手続きにより、VSF モジュールによる並列プロセス間での同期管理を行う。本手法においては、VSF モジュール間でのコミュニケーションのタイミングを除き、VSF モジュールの内部的な処理を自由に実行することが許される。したがって、VSF モジュール開発における自由度が増すとともに、既存のシミュレーション資源の利用も容易となる。

5. コミュニケーションメカニズム

OSAViS では、様々な VSF モジュールが VSF サービスの授受を行い、シミュレーションを実行する。VSF サービスの要求や提供は、VSF モジュール間でのコミュニケーションメッセージの授受という形で行われる。そこで本節では、OSAViS における VSF モジュール間でのコミュニケーションを実現するための仕組みについて説明する。

5.1 VSF サービス指定によるコミュニケーション

OSAViS においては、様々な VSF モジュールがネットワーク接続された複数の計算機上に分散して配置され、必要に応じてその

構成を変える。したがって、こうした構成の変化に影響されない、VSF サービスの授受の仕組みが必要となる。このような課題に対して、オブジェクト指向に関する業界団体である OMG(Object Management Group) では、CORBA とよばれる分散オブジェクト間でのメッセージ交換に関する仕様の制訂作業が進められている⁶⁾。CORBA では ORB とよばれる仲介者がオブジェクトの識別子とその位置情報を維持管理することでオブジェクト間のメッセージ交換を実現する。

そこで OSAViS では、同様の仕組みで VSF サービスの授受を実現する。ただし、CORBA の場合は、識別子により陽にオブジェクトを指定し、処理を要求するのに対し、OSAViS においては、VSF モジュールに対して VSF サービスの要求を行う場合、VSF モジュールを陽に指定せず、VSF サービスのみを指定してメッセージを送信し、VSF サービスマネージャがそのメッセージに適切な VSF モジュールを割り当てるという手続きをとる。ここで、VSF サービスマネージャとは、仮想ショップフロアを構成している VSF モジュールと VSF モジュールが提供可能な VSF サービスに関する情報を持ち、それらの対応づけを行うモジュールである。以上の仕組みにより、モジュール開発者が任意の VSF サービスを VSF モジュールとして実装することが可能となり、モジュール開発における自由度が増す。

次項では、以上で示した VSF サービスの指定によるコミュニケーションの手続きについて説明する。

5.2 コミュニケーションの手続き

VSF モジュール間でやり取りされるメッセージは以下に示す 4 つの情報を含んでいる。

- (1) メッセージ ID; メッセージを特定するための情報。以下で説明する手続きにより送り手の情報も含む。
- (2) サービスコード; VSF サービスの種類を表すコード。これにより、メッセージの送信すべき相手を VSF サービスマネージャが特定する。
- (3) サービス制御情報; VSF サービスに関する細かな指定をするための情報。
- (4) データ本体; 具体的に送りたい情報の内容。

図 3 に示す例により、メッセージに含まれるメッセージ ID およびサービスコードを用いたメッセージ授受の手続きについて説明する。この例では、“hosta”のマシン上のモジュール番号“5555”の VSF モジュール A が、サービスコード“service1”の VSF サービスを要求し、そのサービスを提供できる VSF モジュール B が“hostb”上の“4444”である場合を想定する。

OSAViS において、VSF モジュールは、モジュール外部に対する入出力機能を持つサービスインタフェースを備えており、このインタフェースを介して VSF サービスの授受が行われる。サービスインタフェースとは、VSF モジュールを OSAViS における開放型仮想ショップフロアの要素として利用する上で、モジュール間での確実な相互接続を実現するために標準化されたインタフェースである。以下では、このサービスインタフェースと VSF サービスマネージャを用いたコミュニケーションの流れについて述べる。

- (1) VSF モジュール A は、モジュール内部でローカルにメッセージを区別する ID 番号(図 3 の例では“123”)を付加し、サービスコードを“service1”として、出力部にメッセージを送る。
- (2) VSF モジュール A のサービスインタフェースの出力部は、サービスコードを VSF サービスマネージャに送り、そのサービス

Fig.3 Example of communication

を提供できる VSF モジュールのホスト名とモジュール番号の情報を得て、そのモジュールに対して、VSF モジュール A のホスト名とモジュール番号をメッセージの ID に付加し、送信する。

- (3) VSF モジュール B の入力部は常に待ち状態になっており、VSF モジュール A からの接続要求を受けて、メッセージを受け取る。
- (4) メッセージを受け取った VSF モジュール B は、その要求に従い、様々な処理を行ってその結果をメッセージとして、受け取ったメッセージの ID と要求に対する返事であることを示すサービスコード "serve" を付けて出力部に送る。
- (5) 出力部は、受け取ったメッセージの ID にしたがって、VSF モジュール A に対して、そのメッセージを送る。このとき、メッセージの ID に付加されているホスト名とモジュール番号は VSF モジュール B のものに書き換えられる。

以上の手続きにより、各々の VSF モジュールはシステム内にどのようなモジュールがどのハードウェアに存在するかということとを考慮することなく、必要とする VSF サービスを指定するだけで、要求した VSF サービスの提供を受けることが可能となる。

6. システムの実装

著者らは、OSAVis の有効性を検証するため、そのアーキテクチャに基づくシステムである Open-VirtualWorks の実装作業を進めている。本節では、これまでに述べた手法の具体的な処理手順を AGV の挙動シミュレーションの例を用いて説明する。

6.1 システムの構成

公開データ管理、同期管理、モジュール間コミュニケーションを実現するために、前述のデータオブジェクトマネージャ (Data Object Manager)、同期管理モジュール (Synchronization Control Module)、VSF サービスマネージャ (VSF Service Manager) を含め図 4 に示す 5 種の VSF モジュールを開発し、システムを構築した。

なお、例題で用いられているデータオブジェクトのタイプは、生産設備や施設の形状を定義するソリッドモデルデータタイプおよびショッフロアの可視化に利用される色を定義するカラーデータタイプがある。また、データオブジェクトの利用者として以下に示す 2 種の VSF モジュールを用意した。

Fig.4 Structure of pilot system

(1) AGV モジュール (AGV Module)

無軌道搬送台車 (AGV) の挙動を計算する VSF モジュールであり、データオブジェクトの中の一つである AGV のソリッドモデルデータオブジェクトの所有者モジュールである。

(2) 可視化モジュール (Viewer Module)

ソリッドモデルデータオブジェクトおよびカラーデータオブジェクトのユーザモジュールであり、ショッフロアの 3 次元 CG 表示を行う。

以上の VSF モジュールを用いることにより、本システムでは、ショッフロアに存在する AGV の挙動に関する並列分散シミュレーションを実現する。

6.2 シミュレーションの実行プロセス

本システムにおけるシミュレーションの実行プロセスを以下に示す (図 5 参照)。

- (0) シミュレーション開始前の初期準備として、可視化モジュールが公開データを参照し、ショッフロアの初期状態の 3 次元 CG 表示を行う。このとき、可視化モジュールは、参照したデータオブジェクトにユーザとして登録される。また、それぞれの VSF モジュールは VSF サービスマネージャにアクセスし、必要としている VSF サービスと VSF モジュールの対応づけを行い、モジュール間コミュニケーションのための準備を行う。
- (1) シミュレーションが開始されると、同期管理モジュールは現在基準時間ユニットを他の VSF モジュールに対して送信する。
- (2) AGV モジュールは、現在基準時間ユニットの告知を受けた後、そのユニットにおける AGV の位置、姿勢を計算し、AGV のソリッドモデルデータオブジェクトの位置変更メソッドおよび姿勢変更メソッドの実行を要求し、データの変更を行う。
- (3) 変更が加えられたデータオブジェクトは、ユーザである可視化モジュールに対して変更の通知を行い、可視化モジュールはそれにしたがって新しい位置姿勢のデータを参照して 3 次元 CG 表示を更新する。
- (4) AGV モジュールはその時刻における処理を終えるとその完了を同期管理モジュールに知らせ、同期管理モジュールはすべてのモジュールの処理の完了を確認した後、現在基準時間ユニットを更新し、以上の手続きを繰り返す。

以上のようにして、シミュレーションに関する一連の処理が行われる。

Fig.5 Simulation process of pilot system

6.3 並列分散シミュレーションの運用

前述の VSF モジュールを用いて並列分散シミュレーションを実行する際に、配慮すべきことがいくつか存在する。本論文で提案する同期管理手法では、モジュール個別の時間軸の設定と利用を認めているため、データオブジェクトの利用に際し、所有者モジュールとユーザモジュールの間で、利用のタイミングや頻度が異なり、常に最新のデータを反映できない場合が生じる。また、VSF モジュールの規模と能力などの様々な要因を考慮して、同一の VSF サービスを提供する VSF モジュールの数やネットワーク上での配置を行う必要がある。

こうした効果的な並列分散シミュレーションの実現のためには、

- (1) 全体として仮想シヨップフロアのシミュレーションに求められる精度や性能の定量的な評価と、その評価に基づく VSF モジュールの適切な選択と調整のための手法の開発
- (2) VSF モジュールをその処理負荷に応じて、動的に利用可能な分散した計算資源に割り当て、並列分散シミュレーションの最適化を行う手法の開発

が求められる。こうしたシミュレーション運用時の手法について、現在、研究を行っている。

6.4 並列分散シミュレーションの評価

並列分散シミュレーションの評価を行うために、以上で述べた VSF モジュールをネットワーク接続された複数の計算機上に実装し、使用する AGV モジュールと計算機への割り当てを変えた 14 通りのケースについて 200 の基準時間ユニットのシミュレーションにかかる時間の測定を行った。なお、ここでは VSF サービスマネージャはシミュレーションの初期段階でのみ利用されシミュレーション実行中には関係ないため考慮していない。また、可視化モジュールはグラフィックスワークステーション上で常に単独に処理を行った。

処理時間の測定結果を図 6 に示し、その考察を以下にまとめる。なお、図 6 の縦軸の記号は、VSF モジュールの種類を示し、一つの枠で囲まれている VSF モジュールが、1 台のコンピュータで実行されていることを表す。

- (1) VSF モジュールの計算機への割り当ての変更に対して、VSF モジュール自体の変更をすることなく対応し、シミュレーションが可能であった。
- (2) 全体として分散処理の効果が認められる。特にモジュールの数が多ければその効果が顕著に現れる。図 6 のケース 2 と 3、4 と 5 および 6 と 7 より、同期管理モジュール (S) およびデータオブジェクトマネージャ (D) の分離が処理時間の短縮に効果的であるといえる。これは、これらのモジュールがすべて

Fig.6 Processing time

の AGV モジュールに関して処理を行うため、システムのボトルネックになるためであると考えられる。

7. おわりに

本論文では、著者らの提案する OSAViS に基づく並列分散シミュレーションを実現するため、複数の VSF モジュールにより利用されるデータの管理手法、並列プロセス間での同期管理手法、および、モジュール間でのコミュニケーションの仕組みについて述べた。さらに、OSAViS に基づくシステムである Open-VirtualWorks の開発を行い、本論文で提案する手法の有効性の検証を行った。その結果を以下にまとめる。

- (1) データオブジェクトを導入することで、データ個別の状態を考慮した細やかな管理と、データ変更の通知など様々な付加処理が可能となった。
- (2) 基準時間軸と個別時間軸の 2 種類の時間軸を導入した同期管理手法を提案した。これにより、VSF モジュール内部の処理の独立性を考慮した柔軟な同期管理が実現した。
- (3) 直接 VSF モジュールを指定せず、VSF サービスを指定するコミュニケーションを実現することで、VSF モジュール開発における独立性と仮想シヨップフロアの構成の変更における適応性が向上した。

今後、より大規模で多様なシヨップフロアに対応するため、様々な VSF モジュールの開発、データオブジェクトの定義と実装を行う必要があると考え、それらに関する研究、開発を行う予定である。

参考文献

- 1) 仲町英治ほか: ヴァーチャルファクトリー - 未来工場への挑戦 -, 工業調査会, (1994).
- 2) M.Onosato and K.Iwata: Development of a Virtual Manufacturing System by Integrating Product Models and Factory Models, Ann. CIRP, **42**, 1 (1993) 475.
- 3) S.Osaki, M.Onosato and K.Iwata: Open System Architecture for Virtual Shop Floor and its Modular and Distributed Simulation System, Rapid Product Development, Proc. 8th ICPE, Chapman & Hall, (1997) 499.
- 4) 尾崎 優, 小野里 雅彦, 岩田 一明: 開放型仮想シヨップフロアの開発 (第 1 報) - 仮想シヨップフロアのための開放型システムアーキテクチャの提案 -, 精密工学会誌, **64**, 9 (1998) 1309.
- 5) S.Fujii et al.: Synchronization Mechanisms for Integration of Distributed Manufacturing Simulation Systems under CIM Environment, Proc. 1st ISICIMS '94, (1994) 268.
- 6) 中川路哲男: 分散情報処理によるオープン化, 精密工学会誌, **63**, 5 (1997) 617.